**结合P2P软件使用Ansible分发大文件**

**一 应用场景描述**

 现在我需要向50+数量的服务器分发Logstash新版本的rpm包，大概220MB左右，直接使用Ansible的copy命令进行传输，命令如下：

ansible all -m copy -a "src=/opt/software/logstash/logstash-agent-2.3.3-fb.centos6.x86\_64.rpm dest=/opt/software/logstash"

在执行的过程中，很快就收到Zabbix网络监控的报警，报警项目就是瞬间流量变化大于5Mbps。同时，有的服务器很快执行完成，有很多出现ssh连接错误，Ansible卡死。

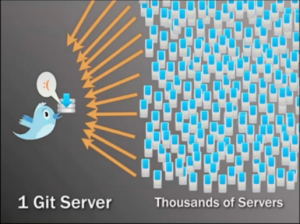
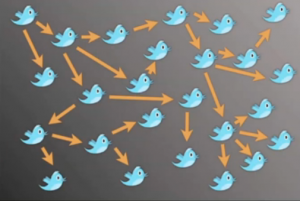
 开启Ansible的pipelining功能依然卡在ssh连接上。分发文件失败！！

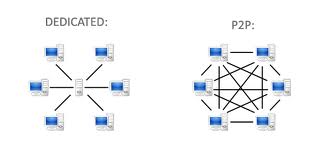
所以，使用Ansible来分发执行命令很快速，但是如果直接使用Ansible来处理稍微大一点的文件分发就是一个很大的问题，即使耗费点时间可以忍受，但是单个分发点的带宽也会直接影响分发效率。

对于大文件分发，首先想到的就是BitTorrent，利用P2P协议实现快速分发，节省带宽，提高效率。

**二 P2P软件介绍**

 这里我们使用Twitter开源的murder。Twitter用它来分发大文件完成代码更新。在早期，Twitter为每天向上万太台的服务器发布代码而头疼，从中央代码服务器向其他成千上万的节点分发代码存在很大瓶颈，因为分发代码的执行时间与需要更新代码的节点成线性关系，节点越多，分发时间越长。为了解决这个问题，Twitter抛弃了以往的集中式架构，转向分布式架构，取名叫murder。使用murder后，他们以前需要40~60分钟的代码发布任务，现在12秒以内就可以完成。



wget https://github.com/lg/murder/archive/master.zip -O murder.zip

unzip murder.zip

cd murder-master

muder是基于BitTornado来实现的。主要有以下几个组件：

**torrent tracker**

        tracker使用murder\_tracker.py运行，tracker实际上就是运行在一台服务器上的单个服务，其他任何成员都要依赖这个tracker。tracker-less disctribution(DHT)目前不支持。tracker存放BitTorrent客户端需要更新状态的路径。

**seeder**

seeder就是存放需要向其他主机分发的文件的服务器。这些文件存放在seeder的一个目录，torrent根据这个目录创建。Murder会将这个目录打包成tgz格式，然后创建一个.torrent文件，这个文件很小，只存放关于这个tgz文件的基本哈希信息。这个.torrent文件让各个peers节点知道他们下载的是什么文件。同时，tracker会保持跟踪有哪些 .torrent文件正在被分发。一旦Murder开始传输文件，seeder服务器是众多主机首先获取文件碎片的地方。

**peers**

  peers就是成百上千需要接收文件的服务器，并且在它们之间可以相互传输文件。一旦一个peer节点下载整个tgz文件完成，它将继续seeding一段时间防止蜜罐效应。

命令行使用murder

**1.开启tracker**

python murder\_tracker.py

muder\_tracker.py实际上调用的这个文件BitTornado/BT1/track.py

track.py有很多参数，如果需要添加参数可以修改muder\_tracker.py

几个重要的参数

--port tracker监听的端口,默认是8998

--dfile  存储近期下载信息的文件

--logfile  tracker日志文件，默认是标准输出

为tracker添加启动脚本/etc/init.d/murder-tracker

#! /bin/sh

#

# Start/Stop murder-tracker

#

# chkconfig: 345 99 99

# description: murder-tracker

# processname: murder-tracker

if [ -f /etc/rc.d/init.d/functions ]; then

. /etc/rc.d/init.d/functions

fi

name="murder-tracker"

murder\_tracker\_bin="/opt/app/murder/dist/murder\_tracker.py"

murder\_tracker\_log="/opt/logs/murder/murder\_tracker.log"

murder\_tracker\_data="/opt/data/murder/tracker\_data"

murder\_user=murder

find\_tracker\_process () {

PID=`ps -ef | grep murder\_tracker | grep python |grep -v $0|grep -v grep|grep -v sh|grep -v root| awk '{ print $2 }'`

}

start () {

getent passwd $murder\_user >/dev/null || useradd -r -s /sbin/nologin $murder\_user

LOG\_DIR=`dirname ${murder\_tracker\_log}`

DATA\_DIR=`dirname ${murder\_tracker\_data}`

if [ ! -d $LOG\_DIR ]; then

echo -e "\e[35mLog dir ${LOG\_DIR} doesn't exist. Creating\e[0m"

mkdir -p $LOG\_DIR

fi

if [ ! -d $DATA\_DIR ]; then

echo -e "\e[35mLog dir ${DATA\_DIR} doesn't exist. Creating\e[0m"

mkdir -p $DATA\_DIR

fi

chown -R $murder\_user:$murder\_user $DATA\_DIR $LOG\_DIR

find\_tracker\_process

if [ "$PID" != "" ]; then

echo -e "\e[35m$name is already running!\e[0m"

else

daemon --user $murder\_user nohup python $murder\_tracker\_bin > /dev/null 2>&1 &

echo -e "\e[35mStarting $name Done\e[0m"

fi

}

stop () {

find\_tracker\_process

if [ "$PID" != "" ]; then

echo -e "\e[35mStopping $name\e[0m"

kill $PID

else

echo -e "\e[35m$name is not running yet\e[0m"

fi

}

case $1 in

start)

start

;;

stop)

stop

exit 0

;;

reload)

stop

sleep 2

start

;;

restart)

stop

sleep 2

start

;;

status)

find\_tracker\_process

if [ "$PID" != "" ]; then

echo -e "\e[35m$name is running: $PID\e[0m"

exit 0

else

echo -e "\e[35m$name is not running\e[0m"

exit 1

fi

;;

\*)

echo -e "\e[35mUsage: $0 {start|stop|restart|reload|status|configtest}\e[0m"

RETVAL=1

esac

exit 0

根据自己情况修改相应的参数

**2.创建torrent文件**

python murder\_make\_torrent.py deploy.tar.gz tracker.twitter.com:8998 deploy.torrent

murder\_make\_torrent.py文件实际上调用的 BitTornado的makemetafile.py 文件

**3.Seed the package播种需要分发的文件包**

python murder\_client.py seed deploy.torrent deploy.tar.gz 172.28.2.200

最后一个参数是本机的IP地址

**4.从所有peers节点获取文件包**

python murder\_client.py peer deploy.torrent deploy.tar.gz 172.28.2.220

**三 使用Ansible执行分发命令**

tracker  172.168.2.171

seeder   172.168.2.179

peers    172.168.2.180~200

murder执行文件目录            /opt/app/murder

tracker和seeder的murder数据目录    /opt/data/murder

peers下载目录               /opt/software/download/

1.在tracker服务器上启动tracker

# ansible 172.168.2.171 -m service -a "name=murder-tracker state=started"

2.在seeder服务器上制作torrent文件并启动seeder

seeder启动脚本/etc/init.d/murder-seeder

#! /bin/sh

#

# Start/Stop murder-seeder

#

# chkconfig: 345 99 99

# description: murder-seeder

# processname: murder-seeder

if [ -f /etc/rc.d/init.d/functions ]; then

. /etc/rc.d/init.d/functions

fi

name="murder-seeder"

murder\_seeder\_data="/opt/data/murder"

murder\_seeder\_log="/opt/logs/murder/murder\_seeder.log"

murder\_seeder\_bin="/opt/app/murder/dist/murder\_client.py"

murder\_make\_torrent\_bin="/opt/app/murder/dist/murder\_make\_torrent.py"

murder\_seeder\_conf="/opt/app/murder/dist/seeder.conf"

deploy\_file=$(awk -F= '/deploy\_file/{print $2}' /opt/app/murder/dist/seeder.conf)

torrent\_file=$(awk -F= '/torrent\_file/{print $2}' /opt/app/murder/dist/seeder.conf)

tracker\_ip=$(awk -F= '/tracker\_ip/{print $2}' /opt/app/murder/dist/seeder.conf)

local\_ip=$(awk -F= '/local\_ip/{print $2}' /opt/app/murder/dist/seeder.conf)

murder\_user=murder

find\_seeder\_process () {

PID=`ps -ef | grep murder\_client|grep seed | grep python |grep -v $0|grep -v grep| awk '{ print $2 }'`

#PID=`ps -ef | grep murder\_client|grep seed | grep python |grep -v $0|grep -v grep|grep -v sh|grep -v root| awk '{ print $2 }'`

}

start () {

getent passwd $murder\_user >/dev/null || useradd -r -s /sbin/nologin $murder\_user

LOG\_DIR=`dirname ${murder\_seeder\_log}`

DATA\_DIR=${murder\_seeder\_data}

if [ ! -d $LOG\_DIR ]; then

echo -e "\e[35mLog dir ${LOG\_DIR} doesn't exist. Creating\e[0m"

mkdir -p $LOG\_DIR

fi

if [ ! -d $DATA\_DIR ]; then

echo -e "\e[35mLog dir ${DATA\_DIR} doesn't exist. Creating\e[0m"

mkdir -p $DATA\_DIR

fi

####### make torrent

python $murder\_make\_torrent\_bin $deploy\_file $tracker\_ip $torrent\_file

#######

chown -R $murder\_user:$murder\_user $DATA\_DIR $LOG\_DIR

find\_seeder\_process

if [ "$PID" != "" ]; then

echo -e "\e[35m$name is already running!\e[0m"

else

nohup python $murder\_seeder\_bin seed $torrent\_file $deploy\_file $local\_ip > $murder\_seeder\_log 2>&1 &

#daemon --user $murder\_user nohup python $murder\_seeder\_bin seed $torrent\_file $deploy\_file $local\_ip > $murder\_seeder\_log 2>&1 &

echo -e "\e[35mStarting $name Done\e[0m"

fi

}

stop () {

find\_seeder\_process

if [ "$PID" != "" ]; then

echo -e "\e[35mStopping $name\e[0m"

kill $PID

else

echo -e "\e[35m$name is not running yet\e[0m"

fi

}

case $1 in

start)

start

;;

stop)

stop

exit 0

;;

reload)

stop

sleep 2

start

;;

restart)

stop

sleep 2

start

;;

status)

find\_seeder\_process

if [ "$PID" != "" ]; then

echo -e "\e[35m$name is running: $PID\e[0m"

exit 0

else

echo -e "\e[35m$name is not running\e[0m"

exit 1

fi

;;

\*)

echo -e "\e[35mUsage: $0 {start|stop|restart|reload|status|configtest}\e[0m"

RETVAL=1

esac

exit 0

启动脚本依赖一个配置文件seeder.conf

# cat /opt/app/murder/dist/seeder.conf

deploy\_file=/opt/data/murder/deploy.tar.gz

torrent\_file=/opt/data/murder/deploy.torrent

tracker\_ip=172.168.2.171:8998

local\_ip=172.168.2.179

将需要的分发的文件打包成deploy.tar.gz

启动seeder

# ansible 172.168.2.179 -m service -a "name=murder-seeder state=started"

3.从seeder获取种子文件，然后分发到peers

# ansible 172.168.2.179 -m synchronize -a "mode=pull src=/opt/software/download/deploy.torrent dest=/opt/software/download"

调用synchronize模块，pull模式就是从远端获取文件到本地，默认是push模式，从本地推送文件到远端

然后将种子文件分发出去

# ansible all -m synchronize -a "src=/opt/software/download/deploy.torrent dest=/opt/software/download"

4.在各个peers端执行下载任务

# ansible all -m shell -a "sh /opt/app/murder/dist/peer\_download.sh"

peer\_download.sh

#!/bin/bash

#this file is used to download bt files

torrent\_file=/opt/software/download/deploy.torrent

download\_file=/opt/software/download/deploy.tar.gz

local\_ip=$(hostname -I|awk '{print $1}')

murder\_client\_bin=/opt/app/murder/dist/murder\_client.py

python $murder\_client\_bin peer $torrent\_file $download\_file $local\_ip

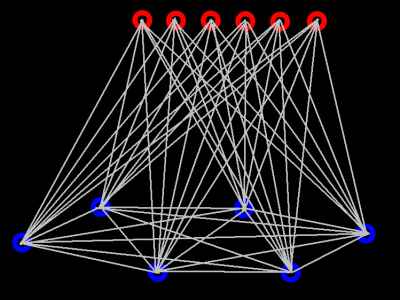
分发完成

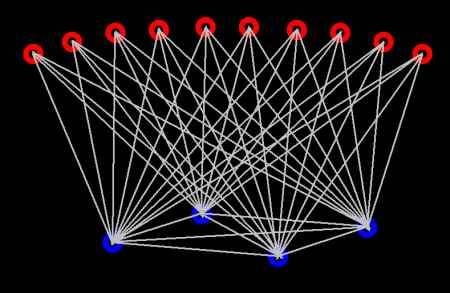
可以将这些步骤写成Ansible playbooks

需要注意一下：

我需要分发的服务器是外网服务器，每台服务器开启了iptables防火墙。总共有60多台服务器同时下载220M左右的压缩包总共花了约20多分钟时间。这个时间有点怀疑，通过再次了解BT原理和查看源代码发现是防火墙设置的问题。BT下载之所以是下载点越多，下载速度越快，是因为各个下载点之间可以交换数据，也就是说需要开启TCP端口用于BT下载。这点在murder的文档中是没有说明的，twitter默认是每台服务器都关闭防火墙，并且是处于一个数据中心的彼此相互信任的内网服务器。murder封装的是BTTornado,代码中默认是启动一个10000~60000范围的随机端口，每个murder peer在下载的同时向其他peers提供下载服务就是通过这个随机端口，如果防火墙全部关闭，这个不成问题，但是如果开启了防火墙这么大的端口范围肯定不行的，就需要自己设置一个防火墙允许的范围。

如果不开端口也是可以上传数据的，但是会影响下载速度，因为其他peer端无法连接到彼此。





有关下载的参数在BitTornado/download\_bt1.py中定义有

和端口相关的参数

('minport', 10000, 'minimum port to listen on, counts up if unavailable'),

('maxport', 60000, 'maximum port to listen on'),

('random\_port', 1, 'whether to choose randomly inside the port range ' +

'instead of counting up linearly'),

这个范围太大，根据自己情况设置小一点，然后让防火墙通行